INDICES Y ANALIZADOR SINTACTICO

Comparativa entre usar o no índices y optimizar los Query

Introducción:

Para poder hacer el análisis que hacemos en esta guía, utilizamos la sentencia EXPLAIN. Esta sentencia nos permitirá obtener información sobre el plan de ejecución de nuestras consultas realizadas contra nuestra la base de datos. De esta forma, podremos analizar este plan de ejecución para saber cómo optimizar la ejecución de dichas consultas.

La sentencia EXPLAIN devuelve una tabla con una serie de filas con información sobre cada una de las tablas empleadas en la consulta a la que acompaña. EXPLAIN se puede emplear en las siguientes consultas: SELECT, DELETE, INSERT, REPLACE y UPDATE. Para emplear esta sentencia, bastará con poner EXPLAIN seguido de la consulta que queremos analizar

Aquí se utilizará el diseño gráfico del Workbench con el objetivo de que se entienda lo mejor posible, pero también se lo puede usar y ver correctamente por la consola de comandos.

Manos a la obra:

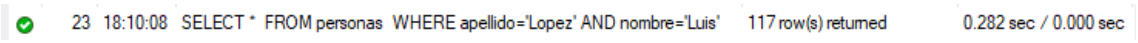
Para esta demostración se usó una tabla “Personas” que contenía 2 columnas (Apellido y Nombre) y a su vez la tabla contaba con 391.820 registros desordenados.

Pongamos el siguiente ejemplo:

*SELECT **

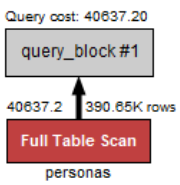
FROM personas

WHERE apellido='Lopez' AND nombre='Luis'



Al ejecutarlo, como vemos en la barra de acción, a la base de datos le tardó 0.28 segundos en poder responder (Si, para complicarlo más, también hay muchos repetidos desordenados)

Pero ahora, ¿Qué sucede con el plan de ejecución?



El Plan de ejecución describe “que es lo que hizo” nuestra base de datos para ejecutar el Query además de darnos algunos valores para poder hablar y pensar en términos de *magnitudes*

En este caso, nuestra base de datos no tuvo más remedio que utilizar un Full Scan, como bien sabemos el Full Scan es algo muy costoso en tiempo para realizarlo, por eso está en rojo (además de que como se lee, tuvo que recorrer los 390.65K (mil) de registros).

El costo de hacer el Query para MySQL en este caso fue de *40637.20*.

También si apoyamos el mouse sobre la acción obtenemos algunos detalles más muy importantes:

```
Key/Index: -

Attached Condition:
  ((`dbname`.`personas`.`nombre` = 'Luis')
     AND (`dbname`.`personas`.`apellido` = 'Lopez'))

Rows Examined per Scan:  390652
Rows Produced per Join:  3906
Filtered (ratio of rows produced per rows examined):  1.00%
   Hint: 100% is best, <= 1% is worst
   A low value means the query examines a lot of rows that are not returned.
Cost Info
  Read: 40246.55
  Eval: 390.65
  Prefix: 40637.20
  Data Read: 518K
```

De aquí podemos sacar 2 cosas importantes:

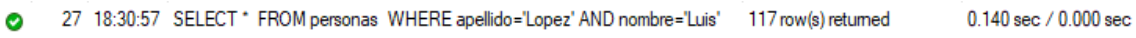
1) (Key / Index : -) Esto significa que no se usó ningún índice para poder ejecutar el Query.

2) (Cost Info:) MySQL le da “valores” a las ejecuciones para que podamos pensar en términos de *magnitudes.*

Ahora, sabiendo esto probemos agregando un Índice a nuestra tabla Personas en la columna Apellidos de la siguiente forma:

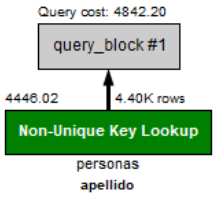
ALTER TABLE personas ADD INDEX (apellido)

Una vez agregado el índice volvemos a realizar la misma consulta:



Y tenemos una gran diferencia en tiempo, pasamos de 0.28 a 0.14 segundos

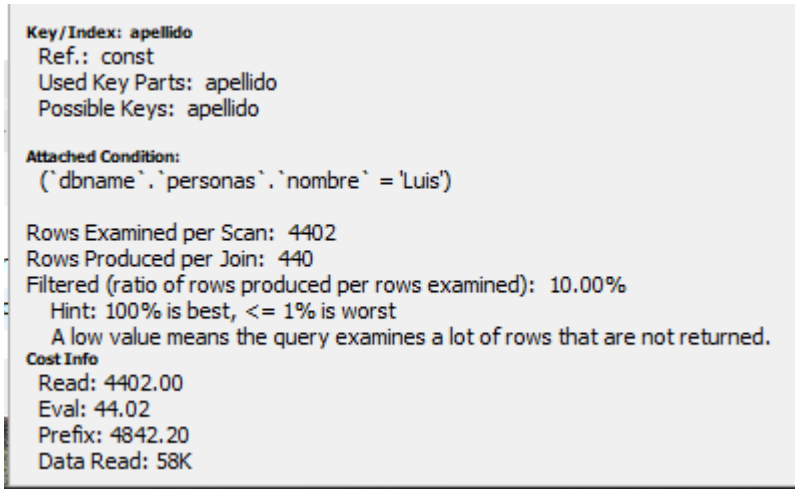
¿Y el plan de ejecución?



No solo ahora está en verde, sino que ya no hizo un Full Scan y por lo tanto tampoco tuvo que recorrer los 390.65K de registros, sino que solamente recorrió 4.40K.

Además, el costo del Query ya no es de *40637.20* como el anterior caso, sino fue algo mucho mas pequeño *4842.20*

Y si apoyamos el mouse nos da más información:



En primer lugar, rápidamente nos damos cuenta que usó el índice que le proporcionamos a la base de datos y, por otro lado, los costos bajaron muchísimo en valores pasamos de tener *518K* de datos leídos a tener solo *52K*.

Este tutorial fue realizado bajo el S.O. Windows 10 y MYSQL 8.0.

El Script "Persona" también será subido para que puedan utilizarlo.